Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

Developing Intelligent Environments with OSGi and JADE

Davide Carneiro[1], Paulo Novais[1], Ricardo Costa[2], José Neves[1]
[1] Department of Informatics, University of Minho, Braga, Portugal
{dcarneiro, pjon, jneves}@di.uminho.pt
[2] College of Management and Technology - Polytechnic of Porto, Felgueiras, Portugal
rfc@estgf.ipp.pt

Abstract. The development of intelligent environments poses complex challenges, namely at the level of device heterogeneity and environment dynamics. In fact, we still lack supporting technologies and development approaches that can efficiently integrate different devices and technologies. In this paper we present how a recent integration of two important technologies, OSGi and Jade, can be used to significantly improve the development process, making it a more dynamic, modular and configurable one. We also focus on the main advantages that this integration provides to developers, from the Ambient Intelligence point of view. This work results from the development of two intelligent environments: VirtualECare, which is an intelligent environment for the monitorization of elderly in their homes and UMCourt, a virtual environment for dispute resolution.

Keywords: Ambient Intelligence, Online Dispute Resolution, Multi-agent Systems, Service-Oriented Architectures, OSGi, Jade.

1 Introduction

Ambient Intelligence is a relatively new field of Artificial Intelligence. In this paradigm, computers are seen as a proactive tool that assists us in our day to day. For the first time, the user in placed at the center of the computer-human interaction, which constitutes a major shift in the traditional paradigm [9]. In fact, in the past we had to move ourselves to the vicinity of a computer in order to interact with it, using old-fashioned interaction means. Now we are interacting with environments that are built on numerous and distributed small computers that communicate, embedded in our common devices.

Evidently, the development of such environments poses complex challenges, namely because these are highly dynamic environments, include heterogeneous devices, are very complex to model, and need to be reliable. In this paper we address some of these challenges by bringing together two fields from computer science: Multi-agent Systems and Service-Oriented Architectures. Multi-agent Systems (MAS) [1] emerged from the combination of Artificial Intelligence with distributed computational models, generating a new paradigm: distributed artificial intelligence. From the Ambient Intelligence point of view a MAS can be defined as a group of entities (software or hardware) which will “read” the environment they are in and take decisions in order to achieve some common goal (like the comfort or safety of

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

persons in the environment) based on knowledge from every agent in the system. In to the so-called Service Oriented Architectures [2], functionalities are provided in the form of technology-independent services, based on three key concepts: the service provider, the service user and the service registry.

In order to develop an intelligent environment that can incorporate insights from these two trends, we consider in this paper two well known technologies: Jade agent platform [3] and OSGi service platform [4]. Jade (Java Agent Development Framework) is a software framework that significantly facilitates the development of agent-based applications in compliance with the FIPA specifications. FIPA (Foundation for Intelligent Physical Agents) promotes standards that aim at the interoperability and compatibility of agents [10]. The use of OSGi (Open Services Gateway Initiative) allows developers to build java applications on a modular basis. The resulting modules are called bundles, which are not only competent to provide services, but also to use services provided from other bundles. In OSGi, a bundle can be installed, started, stopped or un-installed at run-time and without any kind of system reboot, making OSGi-based technologies very modular and dynamic.

Each of these two technologies has already been used successfully to implement a variety of projects in this field following two separate approaches (e.g. [11, 12, 13]). In fact, each one has, as will be seen ahead, characteristics that can be incorporated in intelligent environments, resulting in advantages that are noticed in all the phases of the development, ranging from the specification and design of the system to the actual deployment and use. Moreover, there are approaches that use these two technologies together. [17] for example presents one for using Jade and OSGi, together with a methodology for integration. However, a recent development in these technologies has made this step a much simpler one: the recently released version 3.7 of Jade agent platform, which integrates the OSGi technology. Using this integration it is now possible to run JADE agents inside an OSGi environment, package agents code inside separated bundles, update it by means of the bundle update feature of OSGi and give agents access to all typical OSGi features such as registering and using OSGi services. This constitutes an opportunity that must be exploited in order to simplify the process of developing complex intelligent environments.

In this paper we do not present a new methodology for developing agents nor for developing service-based applications. Instead, we present an approach that builds on this new important development and that can make use of existing approaches for agent and service development individually. As an example, a developer could make use of VisualAgent [18] to easily develop the agents of the system and rely on a Model-Driven Architecture like the one presented in [19] to implement the OSGi services. It results in an iterative and scalable process for developing highly modular applications targeted at virtual environments.

Specification of the Problem

Intelligent environments can be considered in many different domains, namely domestic, medical, legal, public spaces, workplaces, among others. In this paper, given our previous experience, we consider the development of domestic environments for the enhancement of the comfort and security of the user, with an emphasis on home care. In order to highlight the applicability of the presented approach, we will also use as example the development of UMCourt, a platform-

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

independent virtual environment in which parties in dispute find alternatives to a disadvantageous dispute in court. This delimitation of the problem helps to define which devices and services can be considered.

In the case of UMCourt, as it is a virtual environment, it can be accessed from any regular internet-enabled device. The services considered are used transparently be the agents that represent the users and these users do not know with which services they are interacting. They simply request tasks that are delivered to them. In the case of VirtualECare, however, users explicitly interact with different physical devices, with specific capabilities. These devices can be grouped into three categories or layers: computing, communication and interface. Each of these layers has different devices and objectives and only with a close integration of all of them the system can be built.

In the computing layer one can have a multitude of devices which have computing capabilities and that can work independently from each other. In that sense we can consider small and common devices such as the mobile phone, mobile computers, watches, displays, photo machines, televisions or PDAs. In our modern homes, even window blinds, coffee machines or refrigerators have computing and networking capabilities. We can also consider environmental sensors such as temperature, luminosity and humidity sensors, smoke and flood sensors.

Concerning the communication layer, the main challenge comes from the heterogeneity of devices with communication capabilities and the different protocols and communication means that exist. A common home setting nowadays can have several networks, namely Ethernet, Wi-Fi, Bluetooth, power line, among others. There is a whole infrastructure of communicating devices that need to be compatible so that they can be integrated. The interface layer is also a very important part of the system since the devices through which the user interacts with the system will create the image that the user will have of the whole system. This means that this interaction must be very user friendly. In that sense, devices such as video cameras, microphones or touch-screens must be considered. In the opposite side we can enumerate the devices through which the system interacts with the environment, namely the actuators that allow controlling home appliances such as lights, heaters, air conditioning devices, among others.

It is a fact that nowadays a multitude of interesting devices exist that can improve our day-to-day living. However, these are very hard to integrate as each one has different characteristics. To address this problem there are already some defined standards but it necessary that manufacturers follow them and that all of them follow a single one. The solution is then to find effective ways to integrate these different devices by creating a compatibility layer that not only allows the devices to communicate but also makes it easier to develop applications to make use of the services provided by the devices.

2 Nature of the Architecture

Given the already mentioned characteristics of the devices that may inhabit an intelligent environment and given the dynamic nature of the interaction paradigm (e.g. user can interact with many eventually mobile devices, user can start or stop devices,

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

user can incorporate new devices), we can state that an architecture for Ambient Intelligence should be dynamic, modular, expansible, flexible, scalable and compatible [14,15]. In order to implement such an architecture the two previously mentioned technologies will be considered together, joining the advantages of Service Oriented Architectures and Multi-agent Systems in order to build a dynamic environment that can incorporate a multitude of heterogeneous devices.

The Role of Jade. Agent-based technologies have been used for the most different purposes. Basically, this paradigm intends to solve problems through the interaction of simple entities called agents. Each agent has its knowledge about the world and its objectives to accomplish, which may be individual or collective objectives. Likewise, agents may either cooperate or compete in order to achieve their objectives [8]. One of the most interesting research trends in this field is in argumentation theory. In argumentation, agents debate, defend their beliefs and try to convince the other agents into believing the same they do in order to achieve their objectives [5]. Argumentation is also suited for solving conflicts that are usual in these environments. The most common example is a situation in which two agents have conflicting objectives (e.g. maintaining comfort versus saving energy). Besides argumentation, negotiation techniques [6] can and have also been considered to address these challenges.

One important issue here is the one of communication between the agents as these must respect a common standard that ensures that all the agents make use of the same ontology and message syntax. In that sense, the use of FIPA-ACL standard is highly useful. By doing so, many drawbacks concerning communication are solved and the compatibility of the architecture with external agents that respect the defined standard is assured, increasing the expansibility.

Given their features, agents will be used for all the high level decision making processes. It is thus necessary to define agents or groups of agents according to their roles in the architecture. This is, evidently, a task that is domain-dependent. An example focused on a domestic environment is described in section 4.

The Role of OSGi. In a common AmI architecture there is a group of components that must be connected. The objective of using an OSGi platform is to create a compatibility layer that can interconnect all the previously mentioned devices with software components such as databases or external service providers. The approach to be followed will consist in hiding each of these different components behind a different OSGi bundle. The key idea of this approach is to hide the singularities of each device and confine them to the respective bundle. This way, depending on the component being controlled, each bundle will be responsible for the interaction logic and the registration and request of necessary services for the correct execution of the component. Developers of other bundles thus do not need to know the specifications of a given component they intend to interact with. They need only to request the services being provided by the bundle that controls the component. In that sense, OSGi bundles can be seen as black boxes providing services: it does not matter what is inside the box as long as we know how to use the service.

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

3 Developing Intelligent Environments with OSGi and Jade

The roles of OSGi and Jade in the architecture have already been briefly depicted. We will now detail how these two tools can be used together in order to improve the process of developing intelligent environments. As stated before, the MAS is in charge of the high level decision processes, relying on tools like negotiation and argumentation to take globally optimum solutions. Additionally, OSGi is used to build a service layer that ensures the compatibility between all the different components of the architecture.

There are four main components in the architecture that allow a logical high level organization and a modular style of development: Jade container (the virtual location where the agents execute), Jade platform (may hold several Jade containers), OSGi bundle (able to provide and to use services from other bundles) and OSGi platform.
The process of creating an intelligent environment supporting the intended features is organized into a group of sequential and eventual iterative phases, as seen in figure 1.

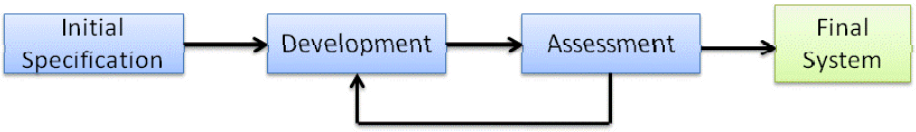

Fig. 1. The three phases of the development process.

Initial Specification. The process starts with the drawing of an initial specification. When defining this, some important factors must be considered such as the target environment of the system, its context of application, the devices that are likely to be used, among others. It is also important here to draw the first sketch of the functionalities of the system in terms of high level components. When drawing this first sketch it is usually useful to have some insights on the final architecture and the organization of the components, so that the first high level components and their organization do not differ significantly from the expected final version. However, as this approach is highly modular, the first sketch of the architecture may be defined without knowing how the final version will be: the intermediary versions can be easily reorganized by changing functionalities (e.g. moving bundles or agents between platforms). This significantly lifts the pressure on the development teams that have to define the architecture of the system without having a clear picture of how the final system will look like, solving one of the biggest challenges in the development of complex systems [7]. Note that in this first stage there is no need to declare which component is of which type, thus giving more freedom to make future changes to the system.

Development. When the process reaches the development phase, the first sketch of the architecture is implemented. This might consist in simply creating high level components according to the initial specification and optionally creating some services with no functionality, only to define the connections between the several components and the way that information will be shared. Note that in this case we define only the several OSGi platforms and the direction of the services that will be used, configuring only high level components.

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

Another important task that can be performed here is to implement simulated bundles instead of "real" ones. Let us call simulated bundle to a bundle that simulates the services it should provide, for example, a bundle that simulates the value of the temperature instead of reading that value from an actual sensor. This is important as it allows developing a prototype version of the system without having all the necessary devices, thus reducing the costs and allowing for more easily and rapidly developing the prototype in initial stages. This means that in an eventual intermediary phase in the process, the system might be constituted only by simulated bundles or a mixture of simulated and real bundles. With OSGi, the whole architecture can be built out of simulated bundles that are then gradually replaced with real ones. The only main concern here is that the bundles that are being replaced have the same name and the same services signature, i.e., for the remaining services they are the same, although the reality is that they provide the same services in different ways.

Assessment. Having implemented this first prototype version, the process moves on to the Assessment phase. In this phase, the system is tested in terms of its efficiency, robustness, usability and scalability. In initial versions, the interest in the assessment is to improve architecture-related parameters. Therefore, one of the important tests to perform is to invoke all the services in order to determine if they are correctly implemented and their signatures respected. It is also vital do evaluate the logical organization of the architecture. This may result in operations like dividing a complex bundle into simpler bundles, dividing an OSGi platform or Jade platform into several ones of the same type if they hold many components, group bundles or agents that are scattered in different platforms into one common platform, among others. In later phases of the development process, the assessment stage may receive as input the experience of interaction with target users. This might result in recommendations for changes suggested by experts in several fields, final users, etc. All these changes are then compiled and passed to a new iteration of the development stage for implementation. This process goes on until a satisfactory architecture is achieved.

By allowing to perform such tasks, this approach allows to develop intelligent environments by following several existing prototyping techniques, namely Throwaway, Evolutionary and Incremental prototyping.

4 Example Settings

4.1 VirtualECare

Having described the nature of the architecture that we achieve with this approach, let us now present an example of how the development process may occur. This example is a simplification of what was the development process of the VirtualECare project [15]. In this example we will assume that we have only temperature and luminosity sensors. According to the proposed process, the first step is to define an initial specification for the system, in terms of the high level roles and functionalities that one intends to implement. In this case we will consider four objectives: we want to monitor the environment in terms of environmental parameters, control the devices

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

present in the environment, incorporate intelligent decision mechanisms and model the user's preferences and needs.

In the first iteration of the development phase, we implement four bundles, each one representing one of the functionalities enumerated before (Figure 2). We also know that the bundle responsible for the decision mechanisms will contain agents so this will be a Jade-OSGi bundle.

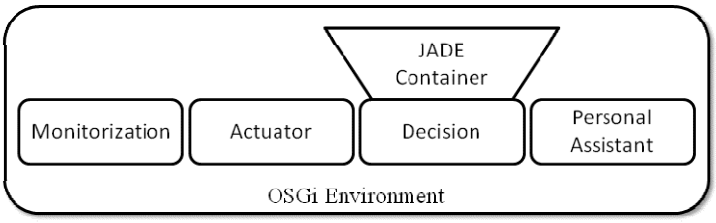

Fig. 2. Result of iteration 1 with four bundles.

This is a rather simple process, consisting in creating only the activators and manifests for each bundle. In the first assessment phase all the bundles are started in order to determine if they have been well defined. In this phase we also define which agents will be necessary in order to be implemented in the next iteration. In this phase we also decide that the Monitorization bundle should be divided into two bundles, one for interacting with real sensors and another one for simulating parameters for which we do not have sensors.

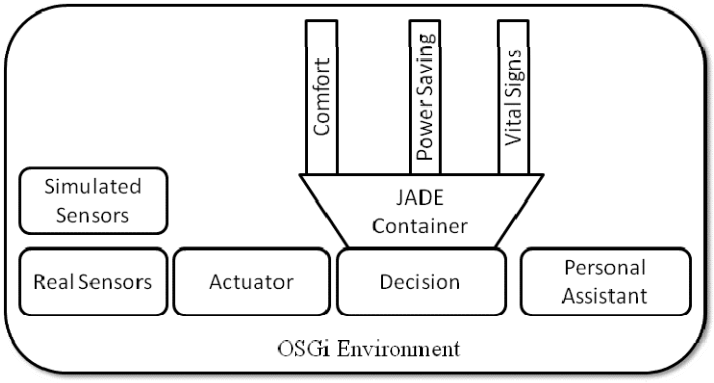

Fig. 3. Result of iteration 2 with five bundles and three agents.

In the second iteration, we implement the decisions that have been taken in the previous one, arriving at an architecture as the one seen in Figure 3. Analyzing this system, we conclude that it makes more sense that the preferences and needs are embodied in two agents so that they also take part in the decision process. Therefore the decision is to move the code from a bundle to two agents. While developing the Real Sensors bundle, it was concluded that, due to the differences in the iteration logic of each type of sensor, this bundle should be divided. The same is decided for the Actuator bundle.

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

In further iterations similar operations can be performed, always without interfering with the components already present. One possible result is the architecture defined in Figure 4, considering a Fault Check bundle that restarts agents that have failed and a database.

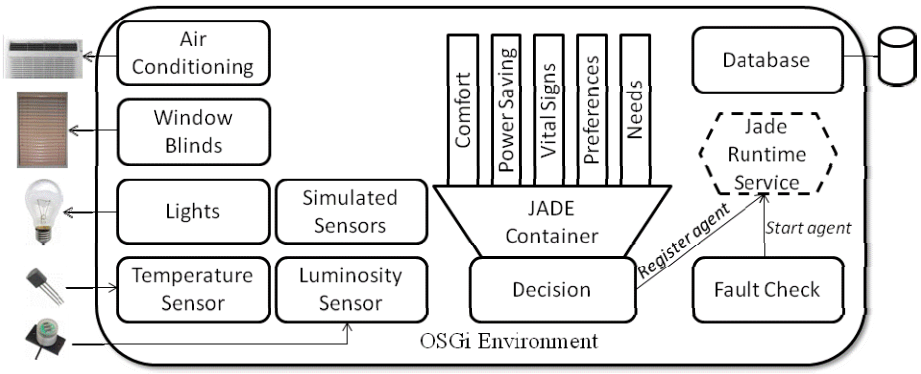

Fig. 4. A simple AmI architecture with 9 bundles, 5 agents and the external components considered.

4.2 UMCourt

UMCourt [16] is the second project in which this approach is being successfully applied. This project aims at the creation of an Online Dispute Resolution platform that uses insights from the Artificial Intelligence field, namely Case-based Reasoning (CBR), to implement a group of services intended to help parties in dispute. These services include the estimation of the most likely outcomes of a dispute, the generation of strategies and solutions, a negotiation environment and general tools for legal documentation management.

Also here, the two technologies have been used with different purposes. Agents are used here in tasks that require significant context information. Examples are the negotiation and Case-based Reasoning modules. In these modules, FIPA-ACL messages are used not only to model information of the legal cases but also to model the control messages that define the negotiation protocol and the CBR process. An example of an ACL Message is shown below.

Example of an ACL message from agent Coordinator to agent Retriever requesting the cases similar to 1263491000923, assuming the default settings.

```
Sender : ( agent-identifier
  :name Coordinator@davide-desktop:1099/JADE
  :addresses (sequence http://davide-desktop:7778/acc ))
Conversation-ID : 1263492569251
Reply-To : Coordinator@davide-desktop:1099/JADE
Ontology : CBR_LABOUR
Content : RETRIEVE_SIMILAR DEFAULT 1263491000923
```

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

OSGi, however, is used differently. It is used to implement the low level tasks that lighten the execution of the agents. Namely, bundles implement behaviors for selecting cases according to given criteria, transparently accessing the database, loading and indexing cases, among others. Adopting this approach has the advantage of decreasing the complexity of the agents by removing these tasks from the scope of the agent. At the same time, it increases code reuse as it is common that different agents make use of some of these tasks that are, this way, encapsulated inside bundles. Figure 5 shows a simplified view of the UMCourt architecture.

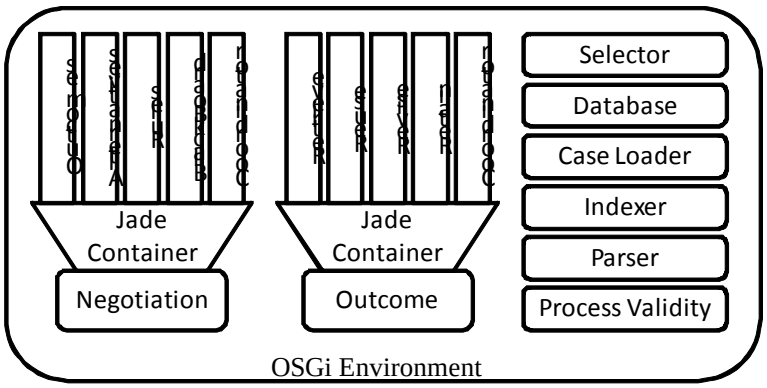

Fig. 5. A simplification of the UMCourt architecture.

5 Conclusions

Jade and OSGi can be used independently to create intelligent environments following two different approaches. However, their integrated use provides a much more powerful solution that can cut development time and requisites. Following the highly modular approach presented here, it is possible to develop these environments gradually, making changes as needed. Hence, it is easy to add new functionalities (in the shape of new bundles or agents), it is easy to rearrange the existing architecture (e.g. sub-dividing or integrating components) always without interfering with the components already present. Functionally, Jade ensures the development of agents and provides all the advantages of a complete messaging service, facilitating the implementation of complex negotiation and argumentation protocols.

OSGi, in the other hand, allows effective integration of very different components, creating a compatibility layer that exposes the functionalities of each component and hides the unnecessary complexity. Concluding, this approach allows to develop intelligent environments following the different prototyping techniques mentioned, fastening the whole process. Moreover, the advantages are present during the development phase and are reflected in the final architecture that is always ready to be improved with a new bundle or agent.

Acknowledgments. The work described in this paper is included in TIARAC - *Telematics and Artificial Intelligence in Alternative Conflict Resolution* Project

Carneiro D., Novais P., Costa R., Neves J., Developing Intelligent Environments with OSGi and JADE, Artificial Intelligence in Theory and Practice III, Bramer M. (Ed), Springer-Verlag, Series: IFIP International Federation for Information Processing, ISBN 978-3-642-15285-6, pp 174-183, (The Third IFIP International Conference on Artificial Intelligence in Theory and Practice (IFIP AI 2010) Brisbane, Australia, September, 2010), 2010.

(PTDC/JUR/71354/2006), which is a research project supported by FCT (Science & Technology Foundation), Portugal.

References

1. Wooldrige, M.: An Introduction to Multiagent Systems. John Wiley & Sons (2002)
2. Perrey, R., Lycett, M.: Service-oriented architecture. In: Applications and the Internet Workshops, Proceedings. vol., no., pp. 116-119, pp. 27-31 (2003)
3. Bellifemine, F., Poggi, A., Rimassa, G.: Developing Multi-agent Systems with JADE, Springer (2008)
4. O.S. Alliance: Osgi service platform, release 3 (2003)
5. Amgoud, L., Parsons, S., Maudet, N.: Arguments, dialogue, and negotiation. In: W. Horn, editor, Proc. 14th European Conf. on AI, pages 338--342, Berlin. IOS Press (2000)
6. Brito, L., Novais, P., Neves, J.: The logic behind negotiation: from pre-argument reasoning to argument-based negotiation. In: PLEKHANOVA, V., ed. lit. - Intelligent agent software engineering, pp. 137-159, London Idea Group Publishing (2003)
7. Edwards, W.K. Grinter, R.E.: At Home with Ubiquitous Computing: Seven Challenges, Proceedings of the 3rd international conference on Ubiquitous Computing, Atlanta, Georgia, USA: Springer-Verlag, pp. 256-272 (2001)
8. Olson, G. M., Malone, T. W., Smith, J. B. (Eds.): Coordination Theory and Collaboration Technology. Mahwah, NJ: Erlbaum (2001)
9. Dix, A., Finley, J., Abowd, G., Beale, R.: Human-computer interaction (3nd ed.). Prentice-Hall, Inc., Upper Saddle River, NJ, USA (2004)
10. FIPA: FIPA ACL Message Structure Specification. Available at http://www.fipa.org/specs/fipa00061/SC00061G.html. <accessed in January, 2010>
11. The Amigo Project: Amigo – Ambient Intelligence for the networked home environment. Short project description (2004)
12. Haigh K., Kiff L., Myers J., Guralnik V., Krichbaum K., Phelps J., Plocher T., Toms D.: The Independent LifeStyle Assistant: Lessons Learned. Honeywell Laboratories (2003)
13. Camarinha, L., Afsarmanesh H.: Virtual Communities and Elderly Support. In Advances in Automation, Multimedia and Video Systems, and Modern Computer Science (2001)
14. Carneiro, D., Novais, P., Costa, R., Gomes, P., Neves, J., EMon: Embodied Monitorization, in Ambient Intelligence European Conference - AmI 2009, Tscheligi M (et al.). (Eds.), LNCS 5859, Springer-Verlag, ISBN 978-3-642-05407-5, pp 133-142 (2009)
15. Costa, R., Novais, P., Lima, L., Carneiro, D., Samico, D., Oliveira, J., Machado, J. and Neves, J., VirtualECare: Intelligent Assisted Living, in Electronic Healthcare, Dasun Weerasinghe (ed.), Springer, Series Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp 138-144, ISBN 978-3-642-00412-4 (2009)
16. Carneiro, D., Novais, P., Andrade, F., Zeleznikow, J., Neves, J., The Legal Precedent in Online Dispute Resolution, in Legal Knowledge and Information Systems, ed. Guido, IOS press, ISBN 978-1-60750-082-7, pp 47--52, (2009)
17. Spanoudakis, N., Moraitis, P., An Ambient Intelligence Application Integrating Agent and Service-Oriented Technologies, in Research and Development in Intelligent Systems XXIV, Proc. 27th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence (AI-2007), pp. 393-398, (2007)
18. De Maria, B.A., Silva, V.T., Lucena, C.J.P., Choren, R.: VisualAgent: A Software Development Environment for Multi-Agent Systems. Proc. of the 19th Brazilian Symposiun on Software Engeneering (SBES 2005), Tool Track, Uberlândia, MG, Brazil, 2005.
19. Model-driven Approach to Real-Time Embedded Systems development (MARTES), 2007. URL: http://www.martes-itea.org